Test 1 (20%)

A. Questions related to Unix. Explain and do the following at the shell prompt.

1. list contents of the directory
2. list files in long format
3. find out the present working directory
4. make a new directory
5. remove an existing directory
6. change UID to that of the superuser
7. peruse the manual page of a command
8. print lines in a file that match a pattern
9. mount a file system
10. mount a file system for read and write
11. unmount a mounted file system
12. change the working directory
13. move files
14. copy files
15. use the command `awk`
16. search for files
17. report file-system disk-space usage
18. reminder service for memorable events
19. create a blank file
20. concatenate files
21. count the number of words in a file
22. look at the boot message
23. print the kernel ring buffer
24. sort lines of a text file
25. redirect a command's output into a file
26. put a regular expression to use
27. use a calculator
28. list all `.c`'s files in the directory
29. prepare for a shutdown
30. bring down the system

Solution.

1. `ls` lists files contained within a specified directory.
2. `ls -l` gives a list of files in the long format.
3. `pwd` stands for 'present working directory'.
4. `mkdir` <*directory*> makes a new directory.
5. `rmdir` <*directory*> removes an existing directory.
6. `su`, which stands for 'superuser'.
7. `man` <*command*> gives the manual page of a command, for instance type `man man` to learn what `man` does.
8. `grep` <*pattern*> <*file*>, for instance `grep while *.c` prints all lines containing the string *while* in all `.c`'s files.
9. `mount` <*file system*>, for example `mount /floppy` should mount a floppy disk
10. `mount -w` <*file system*>, or `mount -o rw` <*file system*>, eg. `mount -w /mnt/hda5` will refer to the contents in the file system table `/etc/fstab` which would say that the hard-disk partition `/dev/hda5` is to be mounted as */mnt/hda5*
11. `umount` <*file system*>, for instance `umount /floppy`. You need to unmount everything you have mounted after you have finished with them. Do `umount /floppy` before removing the disk from the floppy drive.
12. `cd` <*directory*>, for example `cd /etc` changes your working directory to */etc*
13. `mv` <*source*> <*destination*>, for example `mv data data.bak` renames *data* to *data.bak*
14. `cp` <*source*> <*destination*>, for example `cp prog.c ./backup/prog.c` copy *prog.c* into the directory *backup*
15. `awk` is a language for scanning and processing patterns. One example, `awk 'BEGIN { print "hello, world" }'`, does not scan anything but print out a 'Hello World!'.
16. `find /usr -name emacs` finds all files with the name *emacs* residing under */usr* while `find /usr -name emacs*` uses a wild card to include all those names beginning with *emacs*, and therefore includes the whole path if that name happens to be a directory.
17. `du` <*file system*> summarises disk usage of a file system, recursively if it is a directory.
18. To get the list of events for today and tomorrow type `calendar`, or `calendar -f` <*file*> to use *file* instead of files in */etc/calendar* and */usr/share/calendar*. For example, if we have a file called *calfile* whose contents are

```
01/15 test 1
01/16 test 2
01/17 test 3
01/18 test 4
```

then `calendar -f calfile` gives us

```
Jan 16 test 2
Jan 17 test 3
```

19. Use `touch` <*file*> to do this. Actually this command changes the time stamp of a file. But if the file does not exist, it creates first an empty file.
20. `cat` <*files*> concatenate *files* and then print the result on standard output.
21. `wc` <*file*> counts the number of new lines, words and bytes in *file*
22. `dmesg` prints out the contents of the kernel ring buffer, which contains the boot messages.
23. `dmesg`
24. `sort` <*file*> For example, if our *file* contains

```
a z
c s
b u
```

then `sort file` give us

```
a z
b u
c s
```

while to get the result in reverse order type `sort -r file`, which yield

```
c s
b u
a z
```

25. The redirection symbols are `>`, or `>>` The latter does not destroy the existing contents of the file. As an example, `man man > man.manual` types the manual page of `man` and then redirects the text (normally meant for standard output, that is screen) into *man.manual*
26. `cat *.c > ccat` concatenates, that is to say, put together head to tail all C programme files and put the result into *ccat*
27. Try `bc`, type `quit` to end.
28. `ls *.c` should do the job.
29. unmount all file systems manually mounted, for example if we did `mount /floppy` earlier we need to do `umount /floppy` before shutting down.
30. `shutdown` brings the system down and put it into a state where administrative tasks may be performed. `shutdown -h` brings down and halts the system, `shutdown -r` brings down and then reboots the system. `shutdown -h now` brings the system down immediately.

B. Questions related to Emacs. Explain and do the following.

31. move cursor to the beginning of the line
32. move cursor right
33. move cursor left
34. move cursor to the end of the line
35. move cursor up
36. move cursor down
37. move cursor forward one word
38. delete a character
39. erase to end of line
40. read in (insert) the contents of a file
41. insert a line at cursor position
42. set a mark
43. delete to mark
44. yank
45. undo the previous change
46. query and replace
47. discontinue the current action
48. open a file
49. save the current buffer
50. do shell commands from within Emacs
51. put cursor position in the vertical middle
52. scroll forward one page
53. scroll backward one page
54. scroll down one screen
55. scroll up one screen
56. goto the beginning of the buffer (file)
57. goto the end of the buffer (file)
58. search forward for a specified string
59. search backward for a specified string
60. exit from the programme

Solution. In the following `C-` means the control key, `M-` the meta key, `S` the space key and `Sh` the shift key.

31. Type `C-a`
32. Type `C-f`
33. Type `C-b`
34. Type `C-e`
35. Type `C-p`
36. Type `C-n`
37. Type `M-f`
38. Type `C-d` with the cursor on that character.
39. Type `C-k`
40. Type `C-x` followed by `i`, then type in the name of the file including the path if necessary.
41. Type `C-o`
42. Type `C-S`
43. Type `C-w`
44. Type `C-y`
45. Type `C-Sh-_`
46. Type `M%`, then type in what is to be replaced and what takes its place, and then `y` or `n` to either replace or not each instance.
47. Type `C-g`
48. Type `C-x C-f`, then enter the file's name and its path.
49. Type `C-x C-s`
50. Depending on where you want to show the output, `C-u M!` followed by the shell command send the output to a buffer called 'Shell Command Output' whereas `M!` puts your output at the active cursor position.
51. Type `C-l` to place the position where the cursor is half way down the window.
52. Type `C-x ]`
53. Type `C-x [`
54. Type `C-v`
55. Type `Mv`
56. Type `M<`
57. Type `M>`
58. Type `C-s`, then answer what string you want to search for.
59. Type `C-r`, then enter the string you want to find.

60. Type `C-x C-c`

C. Questions related to GCC. Explain and do the following.

61. Write a make file.

Solution.

Consider as an example the following make file where `tst.c` is the name of our source file.

```
cc=gcc
cflags=-c -g -Wall
lflags=-g
libs=-lcurses -ldl -lm
tst : tst.o
        ${cc} ${lflags} $< -o tst ${libs}
%.o : %.c
        ${cc} ${cflags} $<
```

Which says that the C compiler used is GCC. The flag `-c` says that the source codes are to be compiled and assembled, but not linked. The flag `-g` gives information that could be used for debugging. You can use `gdb` to debug using this information. The flag `-Wall` turns on all warnings. Line indentations must be produced by tabs only.

For the following, compile with `gcc -o file file.c`, where `file` is the file's name, then run the executable file.

62. Write a programme to print out some messages on the standard output.

Solution.

Here we use windows I/O instead of regular input and output.

```
/* Extract from Chaucer's Canterbury Tales, Kit Tyabandha, 30 Jan 07 */
#include<curses.h>
#include<stdio.h>
int main(){
  initscr();
  move(7,20);
  printw("But ther been folk of swich condicion\n\
    \t\t That whan they have a certein purpose take,\n\
    \t\t They kan nat stynte of hire entencion,\n\
    \t\t But, right as they were bounden to a stake,\n\
    \t\t They wol nat of that firste purpos slake.");
  move(15,35);
  addstr("Canterbury Tales (c. 1387)");
  move(17,35);
  addstr("Geoffrey Chaucer (c. 1340-1400)");
  box(stdscr, '; '-');
  getch();
  endwin();
  return 0;
}
```

63. Write a programme to calculate $c = a + b$, where a and b are two variables with a specified numerical value.

Solution.

An answer for a minimalist could be the following.

```
1 #include <stdio.h>
2 int
3 main(){
4   int a, b=2, c=3;
5   a =b+c;
6   printf("\n b = %d, c = %d, a = b+c = %d\n\n", b, c, a);
7   return 0;
```

64. Write a programme to calculate $c = a\%b$, where a and b are two specified integral variables.

Solution.

```
1 #include <stdio.h>
2 int
3 main(){
4   int a, b=11, c=3;
5   a =b%c;
6   printf("\n b = %d, c = %d, a = b %% c = %d\n\n", b, c, a);
7   return 0;
8 }
```

65. Write a programme to create and write into a file "Hello World!"

Solution.

```
1 #include <stdio.h>
2 int
3 main(){
4   FILE *fpt;
5   fpt =fopen("tmp.txt", "w");
6   fprintf(fpt, "\n Hello world!\n\n");
7   fclose(fpt);
8   return 0;
9 }
```